

# Corso di Calcolatori Elettronici I

---

## Modi di indirizzamento del processore MC68000 (parte seconda)

Prof. Roberto Canonico



Università degli Studi di Napoli Federico II  
Facoltà di Ingegneria

---

# Utilità di ulteriori modi di indirizzamento

---

---

- Quando un programma deve accedere a dati in memoria può usare:
    - Indirizzamento assoluto: Es. `MOVE $8100,D0`
    - Indirizzamento indiretto: Es. `MOVE (A1),D0`
  - Vantaggio del modo indiretto: l'indirizzo è determinato a runtime, e la stessa istruzione (ad es. all'interno di un ciclo) può operare su dati posti in locazioni diverse
  - Ci sono situazioni in cui un solo grado di libertà attraverso un registro An non è sufficiente
-

# Utilità di ulteriori modi di indirizzamento (cont.)

---

- Ci sono situazioni in cui un solo grado di libertà attraverso un registro  $A_n$  non è sufficiente
  - Esempi
    - Accedere agli elementi di una matrice  $A(i,j)$
    - Accedere ai campi di un array di record
    - Accedere ai campi di un record la cui posizione è determinata a tempo di esecuzione
      - Es. record di attivazione di una subroutine
  - Soluzione:
    - Metodi di indirizzamento che costruiscono l'EA mediante due o più componenti (detti anche *modi con modifica di indirizzo*)
      - $EA = C + R1 + R2 + \dots + Rk$   
con  $C$  = valore costante espresso su  $n$  bit contenuto nella istruzione
    - Il processore MC68000 presenta diversi ulteriori modi di indirizzamento che rientrano in questa categoria
-

# Indexed addressing

---

---

- Detto anche **diretto con registro indice**
  - In generale, l'Indexed Addressing combina due componenti mediante somma, per formare l'EA
    - $EA = C + R = B + I$
    - C è specificato nella istruzione e rappresenta l'indirizzo base (*base address*)  $B$
    - R è contenuto in un registro indice (*index register*) e contiene il valore I da sommare al *base address* per ottenere l'EA (*spiazzamento*)
  - È adatto per accedere ai valori di array e di tabelle
  - Il processore MC68000 non supporta esplicitamente il modo Direct Indexed
-

# Based Addressing

---

- Detto anche **con registro base**
  - Based Addressing è esattamente l'inverso dell'Indexed Addressing
  - Forma l'EA combinando due componenti mediante somma:
    - $EA = C + R = I + B$
    - Il primo componente C, specificato come parte dell'istruzione e quindi costante, assume il significato di *spiazzamento* (*displacement I*)
    - Il secondo componente è contenuto in un registro e rappresenta l'indirizzo base della struttura dati da accedere (*base address B*)
  - È adatto per accedere a campi di record di cui si conosca la posizione relativa ad assembly time, ma non quella iniziale
  - Il processore MC68000 supporta il Based Addressing attraverso il modo ***Indirect with displacement*** d16(An)
  - Es. `MOVE.L 6(A0),D2`
-

# Based Indexed

---

---

- Il modo Based Indexed Addressing forma l'EA combinando una componente costante C e due componenti variabili mediante somma:
    - $EA = C + R1 + R2 = C + B + I$
    - B ha il significato di indirizzo base
    - I ha il significato di *displacement* ed è preso da un registro indice
  - Consente di calcolare a run time sia la posizione iniziale che quella relativa di tabelle ed array
  - Il processore MC68000 supporta lo Short Based Indexed ed il Long Based Indexed
    - Anche detti Indirect with displacement and index
-

# MC68000: Short Based Indexed e Long Based Indexed

---

- **d8(An,Xm)                      Short Based Indexed**
    - d8 spiazzamento costante ad 8 bit in [-128,127]
    - An registro indirizzo
    - Xm è un qualunque registro D0-D7 o A0-A7 che svolge la funzione di registro indice e del quale si prendono solo i 16 bit meno significativi estesi a 32
  - **d16 (An,Xm.L)                      Long Based Indexed**
    - d16 spiazzamento costante a 16 bit in [-32k,32k-1]
    - An registro indirizzo
    - Xm è un qualunque registro D0-D7 o A0-A7 che svolge la funzione di registro indice e del quale si considerano tutti i 32 bit
-

# Relative Addressing

---

---

- “Relative” indica che il calcolo dell’indirizzo è relativo al Program Counter (PC)
  - Questo modo di indirizzamento calcola l’indirizzo effettivo come la somma di un displacement fisso specificato nell’istruzione e del valore corrente del PC
    - $EA = PC + displacement$
  - Fanno spesso uso di displacement piccoli, di 8 o 16 bit, per specificare indirizzi vicini all’istruzione corrente, anziché ricorrere a indirizzi assoluti di 32 bit
  - Il 68000 non consente di utilizzare questi modi di indirizzamento per specificare operandi che potrebbero essere modificati
-



# Relative Indexed Addressing

---

---

- Variante del Relative
  - Funziona come il Based Indexex, ma il base register è sostituito dal PC
    - $EA = PC + X_i + displacement$
  - Può essere usato per saltare ad aree di memoria read-only, contenenti dati o istruzioni
-